The *reservable front end* feature allows you to reserve dedicated access to a Pleiades or Electra compute node for up to 90 days. You can use the node the same way you would use a PFE—to submit jobs and perform other tasks such as post-processing, visualization, and so on—except that the node will be reserved specifically for you (or your group) so you can use all of the node's processor and memory resources. No other users will have access to the node.

Reserving a Dedicated Node

To request a dedicated node, run the "reservable front end" command (pbs_rfe), specifying what type of node you want and for how long. The command makes a PBS advanced reservation for a node, and tells you which node is reserved.

You can submit batch or interactive jobs, check status, and so on, from your reserved node just as you would from a PFE.

Exception: You cannot run an interactive job on Electra nodes from a Pleiades node (or vice versa).

pbs_rfe Command Usage

Run pbs_rfe --help to see the command options and arguments, as shown below:

```
$ pbs_rfe --help
usage: pbs_rfe [-h] [--debug] [--duration DURATION] [--group GROUPS]
               [--model MODEL] [--name NAME] [--starttime STARTTIME]
               [--user USERS] [--version] [-W BILL_GROUP]
               [{request,delete,status,which}] [clue]

...

positional arguments:
  {request,delete,status,which}
                        action to perform
  clue                  node, reservation id, or reservation name

optional arguments:
  -h, --help            show this help message and exit
  --debug, -d           increase debugging verbosity
  --duration DURATION, -D DURATION
                        how long to reserve front-end. Format: [days+]hours
  --group GROUPS, -G GROUPS
                        groups whose members can also use front-end
  --model MODEL, -m MODEL
                        processor type
  --name NAME, -N NAME  name to give reservation
  --starttime STARTTIME, -R STARTTIME
                        when reservation should start. Format as for date(1)
                        command
  --user USERS, -U USERS
                        other userid[s] who can also use front-end
  --version             show program's version number and exit
  -W BILL_GROUP, --group_list BILL_GROUP
                        GID to charge reservation to

In addition to the default "request" action, you can "delete" a
reservation, get the "status" of a reservation, or see "which" node is
assigned to a reservation.

 pbs_rfe delete resv_id       cancel a reservation (only one at a time)
 pbs_rfe status               list your reservations and their statuses
 pbs_rfe which                display the nodes assigned to you
```

The --starttime option allows you to specify when the reservation will start (the default is to start as soon as possible).

The --user option lets you specify other users who will be permitted to log in to your reserved host with you. For example, this might be useful during a shared debugging session. You can also use the --group option to permit all users whose primary group is in the specified list.

If you forget which node is assigned, you can run pbs_rfe which to identify the correct node. For example:

```
$ pbs_rfe which
r617i0n13
```

You can also use the which option in scripts—for example, when you need to learn which node is yours.

pbs_rfe Command Example

This example demonstrates how to use the pbs_rfe command to reserve a Broadwell node for 10 days:

```
$ pbs_rfe --duration 10+ --model bro
```

PBS assigns the r617i0n13 node:

```
$ pbs_rfe --duration 10+ --model bro
Trying to make request for 2 minutes from now
r617i0n13
```

To find out the status of your reservation, run pbs_rfe status. For example:

```
$ pbs_rfe status
Resv name    Resv ID   User      ST        Start / End          Node
-----------  --------  --------  --  ------------ / ------------  ----
FE_zsmith    R3286463  zsmith    CO  Feb 05 15:38 / Feb 15 15:38  r617i0n13
```

Note: You cannot use the node until the reservation starts (a two minute delay, in our example). PBS will send you an email at the start of the reservation period.

When the reservation is ready, the status (ST) will change from CO to RN.

Ending the Node Reservation

When you are finished with a reserved node, you should delete the reservation using the pbs_rfe delete command. For example:

```
$ pbs_rfe status
Resv name    Resv ID   User      ST        Start / End          Node
-----------  --------  --------  --  ------------ / ------------  ----
FE_zsmith    R3286638  zsmith    CO  Feb 09 10:00 / Feb 10 10:00  r601i2n6

$ pbs_rfe delete R3286638
R3286638.pbspl1.nas.nasa.gov deleted.
```

Setting Up the Node

While you can simply SSH into the reserved node from a PFE to run commands, the node is more useful if you start a screen or VNC session there. This allows your sessions on the node to survive even if the PFE you logged into crashes.

Note: This section describes how to set up VNC on the reserved node. For information on using the screen tool, see **man screen** or **info screen**.

VNC Setup

If you want to use a graphical user interface (GUI) to run commands on your node, you'll want to use VNC to set that up. We recommend reading the article VNC: A Faster Alternative to X11 to learn more about VNC and how to establish a VNC session. However, using VNC on a reserved node is simpler than described in that article, once a few preliminary steps are accomplished. These steps are described below.

Before You Begin: Set up SSH passthrough.

Complete these steps:

1. If you do not have the **$HOME/.vnc/xstartup** file, create one on a PFE by following steps 1 and 2 in VNC: A Faster Alternative to X11.
2. Using ssh, log into your reserved node and start vncserver. Do not use the -localhost option.

   ```
   $ ssh r601i1n3 vncserver
       New 'r601i1n3:1 (zsmith)' desktop is r601i1n3:1
       Starting applications specified in /u/zsmith/.vnc/xstartup
       Log file is /u/zsmith/.vnc/r601i1n3:1.log
   ```

3. Create a network tunnel from your workstation to the reserved node. The following command logs you into a PFE with a tunnel that leads to the node:

   ```
   $ ssh -L 5901:node_name:5901 pfe
   ```

   where node_name is the name of the reserved node (r601i1n3, in our example). Note that since you are the only user on your node, the 5901 parts of the command line are constant, unlike when you run vncserver on a PFE node.
4. Start your VNC client on your workstation and connect to localhost:5901. You will be connected to your graphical environment on your reserved node.

VNC Setup for Sharing with Multiple Users

If you share the reservation among multiple users, each with their own VNC session on the reserved node, then users after the first need to use a slightly modified ssh tunnel:

```
$ ssh -L 5901:node_name:59xx pfe
```

where the *xx* in 59*xx* changes for each user. That is, when a later user starts vncserver, the first line of the response will look

something like this:

```
New 'r601i1n3:2 (username)' desktop is r601i1n3:2
```

The number at the end of the line, after the colon, should be added to 5900. In the example above, the number is 2, so the ssh command becomes:

```
$ ssh -L 5901:node_name:5902 pfe
```

Limitations

- The compute node reserved via pbs_rfe does not have the PBS_NODEFILE environment variable set by default. If you plan to run any application that relies on having PBS_NODEFILE set, you will have to set it manually. For example, if you try to run an MPI job on the node without presetting PBS_NODEFILE, you will get the following error message:

  ```
  name_of_compute_node> mpiexec -np X a.out
  Not invoked from a known Work Load Manager:
  o For PBS   : PBS_NODEFILE is not set.
  o For SLURM : SLURM_JOB_ID is not set.
  mpiexec.real error: Aborting.
  ```

  The solution is to create a file, for example a file named "*hfile*", and put the name of your reserved node in it as many times as you want MPI ranks. Then, set the PBS_NODEFILE environment variable to the path to *hfile*.

  ```
  (for bash)
  export PBS_NODEFILE=/path/to/hfile

  (for csh)
  setenv PBS_NODEFILE /path/to/hfile
  ```

- Compute nodes do not have all of the software packages that are installed on the PFEs. You might need to load the current pkgsrc module to get access to some commands. See Using Software Packages in pkgsrc for more information.
- /tmp on a node is small and takes memory away from programs.
- The reserved nodes don't have direct network access to hosts outside Pleiades and Lou. Use your PFE login to transfer files, to outside hosts.
- The cron daemon does not run on reserved nodes. Again, use a PFE for that type of activity.
- While reservations are allowed for up to 90 days, a reserved node might need to be rebooted sooner than that (e.g., for system patches), or you might run the node out of memory. You'll still have the node reserved after the reboot, but anything you were doing will be lost. Please be sure to save your work frequently.
- Currently, each user is limited to only one node reserved at a time.

Account Charging

Reserved nodes are charged Standard Billing Units (SBUs) at the same rate as PBS jobs, whether you are actively using the node or not. As of June 2018, it costs about 44 SBUs per day for a Sandy Bridge node, 61 for Ivy Bridge, 80 for Haswell, 97 for Broadwell, and 153 for Skylake. Reservations are billed daily. Be sure to delete your reservation when you no longer need it.

To reduce costs, a few users can share the same reservation. (See the `--user` and `--group` options above, and Sharing with Multiple Users.

The PBS batch scheduler supports a feature called shrink-to-fit (STF). This feature allows you to specify a range of acceptable wall times for a job, so that PBS can run the job sooner than it might otherwise. STF is particularly helpful when scheduling jobs before an upcoming dedicated time.

For example, suppose your typical job requires 5 days of wall time. If there are fewer than 5 days before the start of dedicated time, the job won't run until after dedicated time. However, if you know that your job can do enough useful work running for 3 days or longer, you can submit it in the following way:

qsub -l min_walltime=72:00:00,max_walltime=120:00:00 *job_script*

When PBS attempts to run your job, it will initially look for a time slot of 5 days. When no such time slot is found between now and the dedicated time, it will look for shorter and shorter time slots, down to the minimum wall time of 3 days.

If you have an existing job that is still queued, you can use the qalter command to add these min_walltime and max_walltime attributes:

qalter -l min_walltime=*hh:mm:ss*,max_walltime=*hh:mm:ss job_id*

You can also use the qalter command to change the wall time:

qalter -l walltime=*hh:mm:ss job_id*

Note: For jobs on Endeavour, be sure to include both the job sequence number and the PBS server name, pbspl4, in the *job_id* (for example, 2468.pbspl4).

If you have any questions or problems, please contact the NAS Control Room at (800) 331-8737, (650) 604-4444, or by email at support@nas.nasa.gov.

During job execution, sometimes it is useful to find out the amount of time remaining for your PBS job. This allows you to decide if you want to gracefully dump restart files and exit before PBS kills the job.

If you have an MPI code, you can call MPI_WTIME and see if the elapsed wall time has exceeded some threshold to decide if the code should go into the shutdown phase.

For example:

```
include "mpif.h"

real (kind=8) :: begin_time, end_time

begin_time=MPI_WTIME()
do work
end_time = MPI_WTIME()

if (end_time - begin_time > XXXXX) then
   go to shutdown
endif
```

In addition, the following library has been made available on Pleiades for the same purpose:

```
/u/scicon/tools/lib/pbs_time_left.a
```

To use this library in your Fortran code, you need to:

1. Modify your Fortran code to define an external subroutine and an integer*8 variable

   ```
   external pbs_time_left
   integer*8 seconds_left
   ```

2. Call the subroutine in the relevant code segment where you want the check to be performed

   ```
   call pbs_time_left(seconds_left)
   print*,"Seconds remaining in PBS job:",seconds_left
   ```

   Note: The return value from pbs_time_left is only accurate to within a minute or two.

3. Compile your modified code and link with the above library using, for example:

   ```
   LDFLAGS=/u/scicon/tools/lib/pbs_time_left.a
   ```

Many jobs running across multiple nodes perform setup and pre-processing tasks on the first node, then launch computations across many nodes, and finish with post-processing and cleanup steps running only on the first node. The post-processing and cleanup steps may take minutes or even hours, during which the remaining nodes in the job are idle. You can use the pbs_release_nodes command to release these idle nodes so that you are not charged for them, and they can be made available to other jobs.

Some users have already taken steps to minimize the number of idle nodes by separating their workflow into multiple jobs of appropriate sizes. The downside to this approach is additional time waiting in the queue for these multiple jobs. Using pbs_release_nodes may allow these jobs to be combined, reducing the amount of wait time in the queue.

Node Tracking and Job Accounting

When you use the pbs_release_nodes command, the released nodes are removed from the contents of $PBS_NODEFILE. The job accounting system tracks when the nodes are released, and adjusts the amount of SBUs charged accordingly.

Running the pbs_release_nodes Command

You can either include the pbs_release_nodes command in your job script or you can run it interactively, for example, on a Pleiades front-end system (pfe[*20-27*]). If you run it in your job script, you can use the $PBS_JOBID environment variable to supply the *job_id*. You can use pbs_release_nodes multiple times in a single job.

There are two ways to use the pbs_release_nodes command. In both cases, the first node is retained.

- **Keep the first node and release all other nodes**
  If you want to retain the first node only, run the command as follows to release the remaining nodes:

  pfe21% pbs_release_nodes -j *job_id* -a

- **Release one or more specific nodes**
  If you want to retain one or more nodes in addition to the first node, run the command as follows to release specific idle nodes:

  pfe21% pbs_release_nodes -j *job_id node1 node2 ... nodeN*

To find out which nodes are being used by a job before you run the pbs_release_nodes command, use one of the following methods:

- **For interactive jobs:** Run qstat -f *job_id*. In the output, look for the nodes listed under the exec_vnode job attribute.
- **For job scripts:** Run cat $PBS_NODEFILE. The output provides a list of the nodes in use.

Note: $PBS_NODEFILE may have duplicate entries for nodes, depending on how resources were requested. See **man pbs_resources** for details.

Sample Job Script with pbs_release_nodes Command

The following sample job script keeps the first node and releases all other nodes before running post-processing steps:

```
#PBS -S /bin/csh
#PBS -N cfd
#PBS -l select=32:ncpus=8:mpiprocs=8:model=san
#PBS -l walltime=4:00:00
#PBS -j oe
#PBS -W group_list=a0801
#PBS -m e
module load comp-intel/2020.4.304
module load mpi-hpe/mpt
cd $PBS_O_WORKDIR
mpiexec dplace -s1 -c 4-11 ./grinder < run_input > intermediate_output
# Before executing some post-processing steps that will run only on the
# first node of the job you can release all other nodes
pbs_release_nodes -j $PBS_JOBID -a
# Now run post-processing steps
./post_process < intermediate_output > final_output
# -end of script-
```

NEW FEATURE: As this feature is still being tested, there may be issues that arise from time to time. If you experience odd behavior, please submit a ticket to support@nas.nasa.gov describing the problem.

The AWS S3/web caching feature enables the use of PBS directives to temporarily pre-cache AWS S3 or web data, making it available to all users in a common location (/nobackup/s3cache/[...]), before a PBS job starts running.

If the data has already been previously cached, the job will experience no additional time spent waiting; it will be placed in the queue to wait for compute resources as it normally would without the directives. If the data has not yet been cached or the data hadn't been accessed recently and was automatically removed, PBS will download the data for the user before the job is started.

Using the PBS Caching Directives

Before You Begin: To enable the PBS caching directives, you must first specify the PBS resource -l site=s3cache in either the qsub command line or in the PBS script, as follows:

- On the qsub command line: pfe: qsub -l site=s3cache ...
- Inside the PBS script: #PBS -l site=s3cache

Without this step, the #CLOUD directives described below will not be parsed and no data will be downloaded.

PBS Directive to Download AWS S3 Data

The directive to download AWS S3 data uses the following format:

#CLOUD -s3cache=s3:{credential_name}{:shared}{:global}{:requester}{:regex}//bucket/prefix/object

Note: A trailing '/' is required when the item referred to is a bucket or prefix/folder. When operating on a bucket or prefix, all objects are downloaded recursively.

The optional arguments marked by {} are:

credential_name
: The name of the profile in the user's ~/.aws/credentials file used to download the data from AWS.

shared
: Indicates that the user wishes the data to be read only to everyone else in the group this job was submitted under as well as just the user. The project ID (GID) used is the user's default GID or the one given by the PBS directive -W group_list=gid. Ignored when no user credentials are given.

global
: Indicates that the user wishes the data to be read only by all users. Ignored if no user credentials are given and the bucket is not covered by NAS.

requester
: Indicates that the S3 bucket is set to requester pays. User credentials are required in such cases unless the bucket is one NAS has pre-authorized the user to access using NAS funds. User credentials if given are used over NAS credentials in cases where both exist.

regex
: The process currently supports regex style wildcards in the right most field of the bucket/folder/object path. Regex expansion in the full path (aside from bucket names) is planned. Note this is only Python regex rules, and not glob'ing or shell wildcard expansion. Without this option the path is treated as a literal string. See below for examples.

PBS Directive to Download HTTP Data

Note: The following directive is too long to be formatted as one line, so it is broken with a back slash (\).

The directive to download HTTP data uses this format:

#CLOUD -s3cache=https:{username}:{password}{:digest_auth}{:shared}\
{:global}//*www.somedomain.com/some/path/to/data*

The stored location for web data is in the form:

/nobackup/s3cache/*www.somedomain.com/some/path/to/data*

The optional arguments marked by {} are:

username
: HTTP 401 Basic Auth username

password
: HTTP 401 Basic Auth password

digest_auth
: boolean, if set uses HTTP Digest Auth instead of HTTP Basic Auth (cleartext)

This has a side effect: it limits what is acceptable as a password. If this becomes a problem, you can try using the shared or global options (described in the previous section).

Note: Domains for HTTP caching must be whitelisted by NAS administrators. The nasa.gov domain is always permitted. Any other

domains require pre-authorization. Please submit a support ticket to support@nas.nasa.gov mentioning cloud support and the domain name(s).

Example PBS Directives

Note: Some of the directives shown below are too long to be formatted as one line; these are broken with a back slash (\).

The following PBS directive would copy data from S3:

```
#CLOUD -s3cache=s3://noaa-goes17/ABI-L2-SSTF/2019/358/21/OR_ABI-L2-SSTF-\
M6_G17_s20193582100338_e20193582159403_c20193582204141.nc
```

By default, the data would be placed in a subdirectory at /nobackup/s3cache/. In this case, the data would be stored here:

```
/nobackup/s3cache/noaa-goes17/ABI-L2-SSTF/2019/358/21/OR_ABI-L2-SSTF\
-M6_G17_s20193582100338_e20193582159403_c20193582204141.nc
```

The following PBS directive would copy data from a public website:

```
#CLOUD -s3cache=https://www.ncei.noaa.gov/pub/data/ghcn/daily/grid/years/1952.tmax
```

By default, the data would be stored here:

```
/nobackup/s3cache/www.ncei.noaa.gov/pub/data/ghcn/daily/grid/years/1952.tmax
```

Note: The downloaded data is read-only for all NAS users by default. Options can be given to change the defaults.

Examples of Regex Usage

Suppose here are two objects in a bucket/folder that are named as follows:

- LT05_L2SP_109077_19950610_20200913_02_T1_SR_B1.TIF
- LT05_L2SP_109077_19950610_20200913_02_T1_SR_B2.TIF

In this case, the directive to download them both (instead of all objects in that folder) would be:

```
#CLOUD -s3cache=s3:regex//bucket/folder/LT05_L2SP_109077_19950610_20200913_02_T1_SR_B.\.TIF
```

Note: The first '.' indicates any single character after the 'B' is acceptable, the '\' before the '.TIF' indicates that the object has a '.' in name that should not be treated as an expression.

Suppose you want to have multiple folders grabbed in a bucket. Use:

```
#CLOUD -s3cache=s3:regex//bucket/LT05_L2SP_109077_199509.._20200912_02_T./
```

This directive would download the two example folders:

- LT05_L2SP_109077_19950914_20200912_02_T2
- LT05_L2SP_109077_19950930_20200912_02_T1

For more regex details, visit the following website and look under the Metacharacters > Special Sequences > Sets sections: https://www.w3schools.com/python/python_regex.asp

URLs or S3 Buckets: Open Data vs. Credentials Required

Web URLs and some AWS S3 buckets provide access without cost. When you request such data in your PBS script, the data will be made available (read-only) to all NAS users under the /nobackup/s3cache/... directory. For examples of open data available via AWS resources, see https://registry.opendata.aws/.

If you are using a URL or S3 bucket that requires credentials, you must must add them to your PBS directive.

Credentials in directives have two forms:

- For URLS: https:{username:password}//... (standard HTTP 401 authorization)
- For S3 Buckets: S3:{AWS_PROFILE}// (AWS credentials)

If user credentials are included in the directives, the download location changes to /nobackup/username/s3cache and the files are read-only, owned by only that user/primary group.

Most S3 buckets are private or "requestor pays." Private buckets require AWS API keys from the owner of the S3 bucket. NAS will not have this information.

NAS can provide credentials for NAS-owned S3 buckets or "requestor pays" S3 buckets. When using NAS-provided AWS credentials, data is downloaded to /nobackup/s3cache/ but it *not* shared globally; it is shared only to the requesting user/group. You can modify the PBS directive to make the downloaded data shared globally (read-only).

Access to S3 buckets that use NAS credentials requires pre-approval. If you want to download from an NAS S3 bucket or public "requestor pays" S3 bucket, please open a support ticket by emailing support@nas.nasa.gov. Include the benefits from NAS paying for the download costs (and not from your project allocation).

Current list of NAS-sponsored buckets: None.

Caching Limitations

When you use the caching directives, please be aware of the following limitations.

30-Day Time-to-Live for Cached Data

By default, there is a 30-day time-to-live (TTL) associated with all cached data. Daily data cleanup automatically removes data that was last accessed beyond its pre-defined time to live. If normal cleanup is insufficient to bring usage below filesystem or bucket quotas, the time-to-live values are reduced and more data is removed until total storage is below quota values.

Quotas for Cached Data

There are quota limits in place for the total amount of data on disk, as well as total amounts of data stored per bucket. These values can vary based on pre-authorized bucket names as well as on a per-user basis.

There are also limits on the amount of data downloaded by a user in a single S3 directive request. This value is currently a single global value. We are exploring options to have the per-directive limit vary based on bucket, user, or other parameters.

Quota and time-to-live values are configurable by NAS staff. If you run into issues with the default, please open a ticket by emailing support@nas.nasa.gov. Changes may require justification and NAS management approval.

qstat Output for Jobs with Caching Directives

When caching directives are used, PBS qstat command output for a job will include these additional resources_used:

CacheAmountDownloaded
: Number of bytes downloaded.

CacheAmountNotDownloaded
: Number of bytes in the request already in cache.

CacheDownloadCost
: The cost to download the data, zero for web data and free S3 buckets.

CacheWaitTime
: How long the job waited before starting to download data.

CacheDownloadTime
: How long it took to download the data once started.

Note: Currently, only two jobs will be processed at a time.